

CUPRINS

13. Securitatea bazei de date	2
13.1. Administrarea utilizatorilor și a resurselor	5
13.1.1. Metode de autentificare a utilizatorilor.....	6
13.1.2. Administrarea utilizatorilor.....	10
13.1.3. Administrarea parolelor și a resurselor utilizând profiluri.....	14
13.1.4. Informații despre utilizatori și profiluri	20
13.2. Administrarea privilegiilor și a role-urilor	21
13.2.1. Privilegii.....	21
13.2.2. Role-uri	22
13.2.3. Acordarea privilegiilor și a role-urilor.....	25
13.2.4. Revocarea privilegiilor și a role-urilor	28
13.2.5. Informații despre privilegii și role-uri.....	29
13.4. Auditarea	30
13.4.1. Opțiuni de audit.....	30
13.4.2. Mecanisme pentru audit.....	33
13.4.3. Informații despre audit.....	35
Bibliografie	36

13. Securitatea bazei de date

- Securitatea unei baze de date presupune monitorizarea acțiunilor realizate de utilizatori asupra acesteia sau a obiectelor sale.
 - În acest sens, sistemul folosește scheme de obiecte și domenii de securitate.
 - Un utilizator este un nume definit în baza de date care poate accesa obiectele acesteia.
 - Obiectele bazei de date sunt conținute în scheme.
 - Definirea utilizatorilor și a schemelor de obiecte ajută administratorii să asigure securitatea bazei de date.
- Accesul tuturor utilizatorilor la anumite obiecte este reglementat prin acordarea de privilegii și *role-uri*.
 - Un privilegiu reprezintă permisiunea de a accesa un obiect într-o manieră predefinită.
 - Un *role* este un grup de privilegii care poate fi acordat utilizatorilor bazei de date sau altor *role-uri*.
- Atunci când este creat un utilizator, administratorul bazei de date trebuie să definească un domeniu de securitate pentru acesta.
 - Sunt specificate metodele de autentificare, spațiile tabel implicate și cele temporare, profilul pentru restricționarea folosirii resurselor bazei de date, privilegiile și *role-urile* acordate utilizatorului respectiv.

Securitate la nivel de sistem

- Presupune administrarea adecvată a utilizatorilor, alegerea metodelor de autentificare a acestora și stabilirea securității sistemului de operare gazdă.
- Fiecare bază de date are unul sau mai mulți administratori care sunt responsabili pentru asigurarea politicilor de securitate (administratori pentru securitate).
 - Dacă baza de date are dimensiuni reduse, administratorul bazei poate avea și responsabilități de securitate.
- Utilizatorii sunt cei care accesează informațiile din baza de date și de aceea, este deosebit de important modul de administrare al acestora.
 - În general, administratorul pentru securitate este singurul utilizator al bazei care are dreptul de administrare a celorlalți utilizatori (creare, modificare, eliminare).

- Sistemul furnizează mai multe metode de autentificare a utilizatorilor bazei de date, folosind:
 - parole;
 - sistemul de operare gazdă;
 - servicii de rețea sau protocolul *SSL (Secure Sockets Layer)*;
 - autentificare și autorizare de tip *proxy*.
- La nivelul sistemului de operare pe care rezidă *server-ul* și aplicațiile de baze de date trebuie ca:
 - administratorii bazei de date să aibă privilegii de creare, respectiv de ștergere a fișierelor;
 - utilizatorii obișnuiți ai bazei să nu dețină privilegii de creare sau de ștergere a fișierelor utile funcționării bazei de date;
 - administratorii pentru securitate să dețină privilegii de modificare a domeniului de securitate pentru conturile disponibile în sistemul de operare (dacă identificarea *role-urilor* pentru utilizatorii bazei de date *Oracle* se realizează prin sistemul de operare).

Securitate la nivel de date

- Include mecanisme de control al accesului la date și al gradului de utilizare a obiectelor bazei.
 - Se stabilesc utilizatorii care au acces la un anumit obiect și tipurile de acțiuni permise asupra sa.
 - În acest sens, se acordă utilizatorilor bazei de date anumite privilegii sistem și obiect, care pot fi grupate în *role-uri*.
 - De asemenea, pentru fiecare obiect al unei scheme sunt definite acțiunile care vor fi auditate.
- O altă modalitate de stabilire a securității la acest nivel este folosirea vizualizărilor.
 - Acestea pot restricționa accesul la datele unui tabel, prin excluderea unor coloane sau linii.
 - Sistemul permite implementarea politicilor de securitate prin folosirea unor funcții și asocierea acestora la tabele sau vizualizări (*fine-grained access control*).
 - O astfel de funcție generează automat o condiție *WHERE* într-o instrucțiune *SQL* și astfel se restricționează accesul la anumite linii de date.

Securitate la nivel de utilizator

- Există două modalități generale de stabilire a securității la nivel de utilizatori:
 - prin intermediul parolelor;
 - prin acordarea privilegiilor.
- Dacă autentificarea utilizatorilor este administrată de baza de date, atunci administratorii trebuie să dezvolte politici de securitate pentru parole.
 - De exemplu, să impună utilizatorilor bazei de date să-și schimbe parolele la anumite intervale de timp, lungimea parolelor să fie suficient de mare, iar în componența acestora să intre atât litere, cât și cifre.
- Problema administrării privilegiilor pentru diferite tipuri de utilizatori este de o deosebită importanță.
 - Pentru bazele de date cu mulți utilizatori, administrarea privilegiilor este mai eficientă dacă se folosesc *role-uri*.
 - În schimb, atunci când numărul de utilizatori este scăzut, se recomandă să se acorde privilegii explicate pentru fiecare utilizator și să se evite folosirea *role-urilor*.
 - Administratorul pentru securitate trebuie să decidă care sunt categoriile de grupuri de utilizatori și să atribuie *role-uri* fiecărui grup în parte. De asemenea, acesta trebuie să decidă ce privilegii trebuie acordate nominal utilizatorilor.
- Administratorii pentru securitate
 - Trebuie să dezvolte politici de securitate inclusiv pentru administratorii bazei de date.
 - Pentru baze foarte mari, care necesită mai mulți administratori, administratorul pentru securitate trebuie să decidă care sunt grupurile de privilegii de administrare și să le includă în *role-uri* de administrare.
 - Pentru baze de date mici este suficientă crearea unui singur *role* specific și acordarea acestuia tuturor administratorilor bazei.
 - Trebuie să creeze politici de securitate pentru dezvoltatorii de aplicații care se conectează la baza de date.
 - Pentru crearea obiectelor, se pot acorda privilegii direct dezvoltatorilor sau numai administratorilor bazei, care să definească obiectele la cererea acestora.
 - Dezvoltatorii de aplicații sunt singurii utilizatori ai bazei de date care necesită grupuri speciale de privilegii. Spre deosebire de utilizatorii finali, aceștia au nevoie de privilegii sistem specifice activității lor.

- Privilegiul sistem *CREATE* este acordat dezvoltatorilor, pentru ca aceștia să poată crea obiectele necesare în aplicații. Dacă dezvoltatorii de aplicații au privilegii pentru crearea obiectelor, administratorul pentru securitate trebuie să controleze limitele de folosire a spațiului bazei de date pentru fiecare dezvoltator în parte.
- Trebuie să definească proceduri de verificare (audit) pentru baza de date.
 - Se poate impune ca aceste proceduri să fie active doar pentru anumite acțiuni sau se pot face verificări prin selecție aleatoare.
 - Administratorul trebuie să decidă care este nivelul minimal la care se fac aceste verificări.
- În cazul sistemelor mari de baze de date, pe care rulează multe aplicații, trebuie să existe administratori pentru aplicații. Aceștia sunt responsabili pentru:
 - crearea și administrarea obiectelor folosite de aplicațiile bazei;
 - crearea *role*-urilor corespunzătoare aplicațiilor și gestiunea privilegiilor asociate fiecarui astfel de *role*;
 - menținerea și modificarea codului aplicațiilor, procedurilor și pachetelor *Oracle*, dacă acest lucru este necesar.
- De cele mai multe ori, administratorul pentru aplicații este unul dintre dezvoltatorii care analizează și proiectează aplicația.

13.1. Administrarea utilizatorilor și a resurselor

- În funcție de licențele primite pentru sistemul *Oracle*, trebuie limitat numărul de sesiuni concurente și de utilizatori conectați la baza de date.
 - Aceasta se realizează prin setarea parametrilor de inițializare de tip *LICENSE*.
- Licențele pentru folosirea concurrentă limitează numărul de sesiuni care pot fi conectate simultan la baza de date.
 - Numărul maxim de sesiuni concurente (*LICENSE_MAX_SESSIONS*) poate fi precizat înainte de a porni instanța și modificat în timp ce baza de date este pornită.
 - Atunci când această limită este atinsă, sistemul va trimite utilizatorilor un mesaj prin care îi anunță acest lucru.
 - În acest caz, se pot conecta la baza de date numai utilizatorii care au privilegiul *RESTRICTED SESSION*.

- Limitarea numărului de utilizatori restricționează numărul autorizațiilor individuale de folosire a sistemului.
 - Precizarea numărului maxim de utilizatori care pot fi creați în bază se face înainte de pornirea unei instanțe (*LICENSE_MAX_USERS*).
 - Această limită poate fi modificată în timpul funcționării instanței, prin folosirea comenzi *ALTER SYSTEM*.
 - După depășirea ei, nu mai pot fi creați noi utilizatori. În acest caz, sistemul va trimite un mesaj prin care anunță că numărul maxim de utilizatori permisi a fost atins.
- Vizualizarea *V\$LICENSE* din dicționarul datelor permite identificarea setărilor curente privind limitările impuse, numărul curent de sesiuni utilizator și numărul maxim de sesiuni concurente atins de la momentul pornirii instanței.

13.1.1. Metode de autentificare a utilizatorilor

- Pentru a preveni folosirea neautorizată a unui cont de utilizator (*username*), sistemul *Oracle* realizează validarea utilizatorilor prin diferite metode, înainte ca aceștia să inițieze o sesiune de lucru cu baza de date:
 - autentificare prin baza de date, folosind parole;
 - autentificare externă, prin sistemul de operare sau servicii de rețea;
 - autentificare globală, prin protocolul de securitate *SSL*;
 - autentificare *proxy*, dacă utilizatorii se conectează la baza de date printr-un *server* de aplicații.
- În general, este folosită aceeași metodă pentru autentificarea tuturor utilizatorilor bazei de date. Totuși, sistemul *Oracle* permite abordarea tuturor metodelor de autentificare, în cadrul aceleiași instanțe a bazei.
- Sistemul necesită proceduri speciale de autentificare pentru administratorii bazei de date, deoarece ei pot executa operații importante asupra acestora.

Autentificarea prin baza de date

- Administrarea conturilor, respectiv a parolelor și validarea utilizatorilor sunt realizate în întregime de către sistem.
- Pentru fiecare utilizator se definesc parole de acces. Din motive de securitate acestea pot fi stocate în format criptat.

Autentificarea externă

- Conturile utilizatorilor sunt întreținute de sistem, iar administrarea parolelor și autentificarea utilizatorilor se fac printr-un serviciu extern (sistemul de operare sau un serviciu rețea, de exemplu *Oracle Net*).
- Conturile utilizatorilor bazei de date vor fi formate dintr-un prefix urmat de numele conturilor acestora din sistemul de operare.
 - Prefixul este setat prin parametrul de inițializare *OS_AUTHENT_PREFIX*.
 - Dacă valoarea acestuia este modificată, atunci conturile care folosesc vechiul prefix sunt invalide.
 - Valoarea implicită a parametrului este *OPS\$*.
- Un utilizator care încearcă să se conecteze la baza de date *Oracle*, va fi autentificat de sistemul de operare.
 - Dacă în acest sistem utilizatorul are contul *nume*, atunci conectarea la baza de date este permisă doar dacă sistemul *Oracle* conține contul corespondent al utilizatorului la nivelul bazei de date (*OPS\$nume*).
- Beneficiile autentificării prin **sistemele de operare** sunt:
 - utilizatorii se pot conecta la sistemul *Oracle* în mod convențional, fără să folosească un cont și o parolă (de exemplu, un utilizator poate să invoce utilitarul *SQL*Plus*, lansând direct comanda *SQLPLUS*);
 - controlul autorizării utilizatorilor este centralizat în sistemul de operare (în baza de date nu trebuie stocate și administrate parolele utilizatorilor).
- Sistemul *Oracle* poate utiliza un **serviciu de rețea** (*DCE*, *Kerberos*, *SESAME* etc.) pentru autentificarea utilizatorilor. Pentru aceasta trebuie utilizate facilitățile aduse de *Oracle Advanced Security*.
- Serviciul de autentificare externă prin rețea folosește infrastructuri cu chei publice.
 - Elementele fundamentale ale acestora sunt certificatele digitale, autoritatele de certificare și facilitățile de administrare a certificatelor.
 - În funcționarea sistemelor cu chei publice (*PKI*) este necesar un sistem de generare, circulație și autentificare a cheilor folosite de utilizatori.
 - Autoritatele de certificare distribuie certificate de chei autenticate.
 - Certificatul reprezintă o asociere imposibil de falsificat dintre o cheie publică și un anumit atribut al posesorului său. El poartă semnătura digitală a unei autorități de certificare care, în acest fel, confirmă identitatea subiectului.

- Sistemele de autentificare care folosesc infrastructuri cu chei publice eliberează utilizatorilor certificate digitale, pentru autentificarea directă la *server*.
- Infrastructura cu chei publice folosită de *Oracle* are componentele:
 - protocolul *SSL* (independent de platformă și de aplicație, care furnizează servicii de autentificare, compresie de date, criptare și integritate a datelor pentru o serie de protocoale *Internet* la nivel aplicație), pentru autentificarea și gestiunea sigură a cheilor;
 - utilitarul *Oracle Call Interface* și funcții *PL/SQL* pentru folosirea semnăturii digitale și verificarea acestora utilizând certificatul asociat;
 - certificate pentru utilizatori, eliberate de o autoritate de certificare care oferă un nivel ridicat de încredere;
 - portofele virtuale, care conțin cheia privată a utilizatorului, certificatul de autentificare și lista certificatelor de bază prin care utilizatorul obține un nivel ridicat de încredere;
 - *Oracle Wallet Manager*, o aplicație *Java* folosită pentru gestiunea și editarea scrisorilor de acreditare din portofelele virtuale (protejează cheile utilizatorilor, gestionează certificatele *X.509v3* pe *server*-ele *Oracle*, generează perechi de chei publice și private, creează cereri de certificare către autoritatea de certificare, instalează certificate, configerează certificate de încredere, deschide un portofel *Oracle* pentru a accesa un *PKI*, creează portofele care pot fi deschise folosind *Oracle Enterprise Login Assistant*);
 - certificate de tip *X.509v3*, obținute de la autorități de certificare externe sistemului *Oracle*;
 - instrumentul *Oracle Enterprise Security Manager*, pentru gestiunea centralizată a privilegiilor;
 - serviciul *Oracle Internet Directory*, care permite configurarea și administrarea centralizată a utilizatorilor, inclusiv privilegii și atribută de securitate pentru autentificarea acestora cu certificate *X.509*;
 - utilitarul *Oracle Enterprise Login Assistant*, pentru deschiderea sau închiderea portofelului virtual al unui utilizator și activarea sau dezactivarea comunicațiilor unei aplicații securizate prin *SSL*.

Autentificarea globală

- Utilizatorii sunt identificați în baza de date ca utilizatori globali, folosind protocolul SSL.
- Administrarea utilizatorilor se face în afara bazei de date, prin centralizarea acestora într-un director, numit director de servicii (*directory service*).
- *Role-urile* globale sunt definite în baza de date, dar autorizațiile pentru acestea se acordă prin directorul de servicii.
- Avantajul gestionării centralizate constă în posibilitatea definirii de utilizatori și *role-uri* la nivel de companie.
- Singurul dezavantaj este că utilizatorul trebuie recreat în directorul de servicii, pentru fiecare bază de date la care trebuie să aibă acces. Soluția este crearea de utilizatori independenți de schemă, ceea ce le permite acestora să folosească o schemă în comun.
- Procesul de creare a unui utilizator independent de schemă presupune:
 - crearea unei scheme partajate la nivelul bazei de date, folosind comanda `CREATE USER nume_schemă IDENTIFIED BY 'specificație_identificare';`
 - crearea utilizatorilor globali în directorul de servicii și asocierea lor la această schemă.

Autentificarea proxy

- În cazul configurației *multitier*, autentificarea *proxy* presupune verificarea dreptului de acces al *server-ului* de aplicații la baza de date, protejând identitatea și privilegiile *client-ilor* de-a lungul tuturor nivelurilor și verificând doar acțiunile realizate în favoarea acestora. De asemenea, sunt controlate acțiunile pe care *server-ul* de aplicații le inițiază în nume propriu.
- Sistemul *Oracle* oferă două forme de autentificare *proxy*:
 - dacă *client-ul* este o aplicație sau un utilizator global, atunci el va fi autentificat de către *server-ul* de aplicații;
 - dacă *client-ul* este un utilizator al bazei de date, atunci el nu va fi autentificat de către *server-ul* de aplicații (identitatea *client-ilor* și parolele trec prin *server-ul* de aplicații către *server-ul* de baze de date, unde are loc autentificarea);
- Pentru a putea să acționeze în favoarea *client-ului*, *server-ul* de aplicații este autorizat de către administrator.

- Principalul avantaj al arhitecturii *multitier* este acela că permite mai multor *client*-i să aibă acces la *server*-ul de date, printr-un *server* de aplicații, fără să fie necesare conexiuni separate.
- Într-un mediu *multitier* autentificarea se realizează în trei etape:
 - aplicația *client* trimite dovada autenticității către *server*-ul de aplicații (parolă sau certificat);
 - *server*-ul de aplicații verifică autenticitatea *client*-ului și apoi se autentifică și el la *server*-ul de baze de date;
 - *server*-ul de baze de date verifică autenticitatea *server*-ului de aplicații, existența *client*-ului și faptul că *server*-ul de aplicații are dreptul de conectare pentru *client*-ul respectiv.

13.1.2. Administrarea utilizatorilor

- Fiecare bază de date *Oracle* conține o listă de utilizatori. Pentru a accesa baza de date, un utilizator trebuie să execute o aplicație a bazei și să se conecteze la o instanță a acesteia, folosind un cont valid definit în bază.
- Crearea unui utilizator se realizează prin comanda *CREATE USER*.
 - Pentru a avea dreptul de folosire a acestei comenzi este necesar privilegiul sistem *CREATE USER*.
 - În general, administratorul pentru securitate este singurul care are acest privilegiu.
 - Crearea unui utilizator constă în definirea identității sale (nume și parolă), specificarea profilului, a spațiului tabel implicit, a cotei de folosire a spațiului tabel și a spațiului tabel temporar în care sunt create segmentele temporare.
- Forma simplificată a comenzi *CREATE USER* este următoarea:

```

CREATE USER nume_utilizator
  IDENTIFIED
    { BY parolă | EXTERNALLY
    | GLOBALLY AS
      'CN = nume_user, alte_atribute_de_identificare' }
  [DEFAULT TABLESPACE nume_spațiu_tabel]
  [TEMPORARY TABLESPACE nume_spațiu_tabel]
  [QUOTA { întreg [{K | M} ] | UNLIMITED}
    ON nume_spațiu_tabel]
  [PROFILE nume_profil]
  [PASSWORD EXPIRE]
  [ACCOUNT {LOCK | UNLOCK} ] } ;
  
```

- Numele unui utilizator trebuie să fie unic.
- Un utilizator și un *role* nu pot avea același nume.
- Modul de autentificare poate fi realizat prin baza de date (*IDENTIFIED BY parolă*), extern (*IDENTIFIED EXTERNALLY*) sau prin protocolul *SSL* (*IDENTIFIED GLOBALLY AS 'specificație_identificare'*).
- Sirul *specificație_identificare* furnizează un mod de identificare la nivelul directorului de servicii.
- Fiecare utilizator trebuie să aibă asociat un spațiu tabel implicit (*DEFAULT TABLESPACE*) în care sistemul stochează obiectele create de utilizator, dacă nu se specifică un alt spațiu tabel pentru acestea. Spațiul tabel implicit este *SYSTEM*.
- Pentru fiecare utilizator se poate specifica un spațiu tabel temporar. Acest spațiu tabel este folosit pentru stocarea segmentelor temporare care sunt necesare comenzilor *SQL* inițiate de către utilizator. Dacă nu este specificat un spațiu tabel temporar, sistemul alocă în mod implicit utilizatorului spațiul tabel temporar care a fost definit la crearea bazei de date. Dacă nici acesta nu a fost specificat, spațiul tabel temporar implicit este *SYSTEM*.
- Cu scopul de a preveni consumul excesiv al spațiului bazei de date se pot preciza limite de folosire (cote) pentru spațiile tabel la care are acces utilizatorul. Aceste limite sunt specificate prin clauza *QUOTA*. Dacă limita spațiului tabel este 0, atunci utilizatorul nu mai poate crea obiecte noi, iar pentru obiectele existente în spațiul tabel respectiv acesta nu mai poate aloca spațiu suplimentar. Opțiunea *UNLIMITED* presupune folosirea nelimitată a spațiului tabel respectiv.
- Privilegiul sistem *UNLIMITED TABLESPACE* permite utilizatorilor să dețină spațiu tabel nelimitat. Acest privilegiu are prioritate față de toate limitele specificate pentru spațiile tabel alocate utilizatorului respectiv.
- De asemenea, la crearea unui utilizator se poate specifica profilul acestuia. Profilurile facilitează administrarea parolelor și a limitelor de folosire a resurselor. Dacă nu este specificat nici un profil, se asociază unul implicit.
- Clauza *PASSWORD EXPIRE* presupune că parola utilizatorului trebuie schimbată la prima conectare la baza de date.
- Pentru blocarea sau deblocarea contului unui utilizator este folosită clauza *ACCOUNT*, cu opțiunile *LOCK*, respectiv *UNLOCK*.

Exemplul 13.1

- a) Se creează utilizatorul *student1*, identificat prin parola *s1t2u3d*. Utilizatorului i se asociază spațiul tabel *users_tbs* cu posibilitatea de a folosi maxim *5MB* din acesta și spațiul tabel temporar *temp_tbs* din care poate folosi maxim *3MB*. Limitarea folosirii resurselor bazei de date este definită de profilul *user_p*.

```
CREATE USER student1
  IDENTIFIED BY s1t2u3d
  DEFAULT TABLESPACE users_tbs
  QUOTA 5M ON users_tbs
  TEMPORARY TABLESPACE temp_tbs
  QUOTA 3M ON temp_tbs
  PROFILE user_p;
```

- b) Se creează utilizatorul *student2* cu opțiunea de autentificare externă.

```
CREATE USER student2
  IDENTIFIED EXTERNALLY
  DEFAULT TABLESPACE users_tbs
  QUOTA 15M ON users_tbs
  PROFILE user_p;
```

- c) Se creează un utilizator al bazei de date care să fie autentificat extern, prin sistemul de operare. Contul corespunzător de la nivelul sistemului de operare este *student3*.

```
CREATE USER ops$student3
  IDENTIFIED EXTERNALLY
  DEFAULT TABLESPACE users_tbs
  QUOTA 10M ON users_tbs
  PROFILE user_p;
```

- d) Se creează utilizatorul global *student*, având drept atribut de identificare jobul (*JB*), departamentul (*DP*) și specializarea (*SP*).

```
CREATE USER student
  IDENTIFIED GLOBALLY AS
    'JB=programator, DP=informatica, SP=oracle'
  DEFAULT TABLESPACE users_tbs
  QUOTA 20M ON users_tbs
  PROFILE user_p;
```

- Pentru a modifica o opțiune a domeniului de securitate al unui utilizator (modul de autentificare, limitele unui spațiu tabel, revocarea unui spațiu tabel, profilul) este necesar privilegiul sistem *ALTER USER*.

- De obicei, acesta este deținut doar de administratorul pentru securitate.
- Comanda folosită este *ALTER USER*, iar modificările specificate prin aceasta nu afectează sesiunea curentă a utilizatorului.
- Singura opțiune pe care fiecare utilizator o poate modifica pentru propriul cont este parola. Pentru aceasta nu este necesar un privilegiu sistem special, altul decât cel de conectare la baza de date. Comanda folosită are următoarea sintaxă:


```
ALTER USER nume_utilizator IDENTIFIED BY parolă;
```
- Ștergerea unui utilizator implică eliminarea acestuia și a schemei asociate, din dicționarul datelor, ceea ce determină ștergerea imediată a tuturor obiectelor conținute în schema.
 - Dacă se dorește păstrarea schemei unui utilizator, iar acesta trebuie să piardă dreptul de a accesa baza de date, i se va revoca utilizatorului respectiv privilegiul *CREATE SESSION*.
 - Nu se poate elimina un utilizator care este conectat la baza de date.
 - Pentru a putea elimina un utilizator este necesar privilegiul sistem *DROP USER*.
 - Dacă schema utilizatorului conține obiecte, trebuie folosită opțiunea *CASCADE* pentru a șterge atât utilizatorul, cât și toate obiectele asociate, inclusiv cheile externe care referă tabelele deținute de utilizator.
 - Dacă utilizatorul are obiecte asociate și nu este utilizată această clauză, atunci sistemul returnează un mesaj de eroare, utilizatorul nefiind eliminat. Sintaxa comenzi folosite pentru ștergerea unui utilizator este:


```
DROP USER nume_utilizator [CASCADE];
```
- Fiecare bază de date conține un grup de utilizatori numit *PUBLIC*.
 - Deoarece orice utilizator al bazei de date face parte în mod automat din grupul *PUBLIC*, privilegiile și *role*-urile acordate acestui grup sunt accesibile tuturor utilizatorilor.
 - Ca membru al acestui grup utilizatorul poate consulta toate tabelele din dicționarul datelor prefixate de *USER* sau *ALL*.
 - Se pot acorda sau revoca grupului *PUBLIC* orice privilegii sistem, privilegii obiect sau *role*-uri. Din motive de securitate, este recomandabil să se acorde doar privilegiile sau *role*-urile care îi interesează pe toți membrii grupului. Acordarea sau revocarea unor privilegii sistem sau obiect grupului *PUBLIC* poate conduce la recompilarea vizualizărilor, procedurilor, funcțiilor, pachetelor și declanșatorilor.

Grupul *PUBLIC* are următoarele restricții:

- nu se pot asocia cote spațiilor tabel deținute de grup, însă se poate acorda acestui privilegiu sistem *UNLIMITED TABLESPACE*;
- se pot crea legături de baze de date sau sinonime publice (prin comanda *CREATE PUBLIC {DATABASE LINK | SYNONYM}*), acestea fiind singurele obiecte deținute de grupul *PUBLIC*.

13.1.3. Administrarea parolelor și a resurselor utilizând profiluri

- Un profil este un set de limitări de resurse care poate fi atribuit unui utilizator al bazei de date. Fiecare bază *Oracle* permite definirea unui număr nelimitat de profiluri. Acestea trebuie create și administrate doar dacă politica de securitate a bazei de date cere ca utilizarea resurselor să fie limitată. Pentru a putea folosi profilurile, mai întâi se creează categorii de utilizatori similari.
- Profilurile pot fi atribuite fiecărui utilizator în parte (folosind comanda *CREATE USER* sau *ALTER USER*) sau se pot defini profiluri implicate care sunt asociate tuturor utilizatorilor care nu au un profil specific.
- Pentru a crea un profil este necesar privilegiul sistem *CREATE PROFILE*. În momentul creării se pot explicita limitele folosirii unor resurse particulare sau parametrii pentru parole.
- Comanda prin care se creează un profil este următoarea:

```
CREATE PROFILE nume LIMIT
  [SESSIONS_PER_USER valoare]
  [CPU_PER_SESSION valoare]
  [CPU_PER_CALL valoare]
  [CONNECT_TIME valoare]
  [IDLE_TIME valoare]
  [LOGICAL_READS_PER_SESSION valoare]
  [LOGICAL_READS_PER_CALL valoare]
  [PRIVATE_SGA valoare]
  [COMPOSITE_LIMIT valoare]
  [FAILED_LOGIN_ATTEMPTS valoare]
  [PASSWORD_LIFE_TIME valoare]
  [PASSWORD_REUSE_TIME valoare]
  [PASSWORD_REUSE_MAX valoare]
  [PASSWORD_LOCK_TIME valoare]
  [PASSWORD_GRACE_TIME valoare]
  [PASSWORD_VERIFY_FUNCTION {funcție | NULL | DEFAULT} ];
```

- Pentru fiecare parametru care apare în comandă se poate preciza o valoare întreagă sau opțiunea *UNLIMITED*, respectiv *DEFAULT*. Opțiunea *UNLIMITED* indică

posibilitatea de folosire nelimitată a resursei respective. Dacă se folosește *DEFAULT*, atunci limita de folosire a resursei va fi preluată din profilul implicit.

- Fiecare bază de date are un profil implicit. Toate limitările de resurse nespecificate pentru profilurile particulare vor fi setate automat la valorile implicate. La crearea bazei de date *server-ul Oracle* definește profilul implicit *DEFAULT*.
- Sistemul *Oracle* poate autentifica utilizatorii folosind informații stocate în baza de date. Cea mai importantă informație de autentificare o reprezintă parola asociată unui utilizator. Aceasta este criptată și stocată în dicționarul datelor. Utilizatorul poate oricând să-și schimbe propria parolă.
- Pentru a asigura confidențialitatea parolelor, sistemul permite criptarea acestora în timpul conexiunilor din rețea (*client/server* sau *server/server*). Dacă este activată această funcționalitate atât pe mașina *client*, cât și pe *server*, sistemul codifică parolele înainte de a le trimite în rețea, folosind o versiune modificată a algoritmului de criptare *DES (Data Encryption Standard)*.
- Dacă un utilizator introduce parola greșit de un număr specificat de ori, sistemul blochează contul asociat acestuia. În funcție de modul de configurare, contul poate fi deblocat automat, după un interval specificat de timp, sau manual, de administratorul bazei de date. Prin parametrul *FAILED_LOGIN_ATTEMPTS* se precizează numărul de conectări care pot eșua înainte de blocarea contului, iar prin parametrul *PASSWORD_LOCK_TIME* numărul de zile în care acesta va fi blocat.
- Administratorul bazei de date poate bloca manual un cont, folosind comanda *ALTER USER*. În acest caz, contul nu mai poate fi deblocat automat, administratorul trebuind să-l deblocheze explicit.
- Prin parametrul *PASSWORD_LIFE_TIME* se poate specifica durata de viață a unei parole. După această perioadă, parola expiră și trebuie schimbată. La prima încercare de conectare după expirare, apare un mesaj prin care utilizatorul este atenționat că trebuie să-și schimbe parola într-un anumit număr de zile (perioadă în care are încă dreptul de conectare). Această perioadă este precizată prin parametrul *PASSWORD_GRACE_TIME*. Dacă parola nu este schimbată până la terminarea perioadei de grație, atunci contul se blochează automat.
- Se recomandă ca schimbarea parolelor să nu se facă folosind comanda *ALTER USER*, deoarece aceasta nu realizează verificarea completă a complexității parolei. Este

preferabil ca pentru schimbarea unei parole să se utilizeze funcția *OCIPasswordChange()*.

- Pentru a verifica dacă o parolă nou specificată nu a mai fost utilizată anterior este menținută o istorie a parolelor. Prin parametrul *PASSWORD_REUSE_TIME* se poate preciza intervalul de timp (exprimat în număr de zile) în care utilizatorii nu pot reutiliza o parolă. De asemenea, se permite și precizarea numărului de schimbări consecutive ale parolei până se poate reutiliza o parolă anterioară (*PASSWORD_REUSE_MAX*).
- Fiecare parolă trebuie să aibă o complexitate minimă pentru a asigura protecția sistemului împotriva eventualilor intruși care încearcă să o descopere. În timpul execuției *script-ului utlpwdmg.sql*, server-ul *Oracle* creează în schema *SYS* funcția implicită *VERIFY_FUNCTION* care asigură verificarea complexității unei parole.
 - Aceasta verifică dacă parola îndeplinește următoarele condiții:
 - are lungimea de cel puțin 4 caractere;
 - nu coincide cu *username-ul*;
 - include cel puțin o literă, o cifră și un caracter special;
 - nu este un cuvânt simplu din lista de noțiuni interne (de exemplu, numele contului, al bazei de date sau al utilizatorului etc.);
 - diferă de parola anterioară cu cel puțin 3 caractere.
- Rutina de verificare a complexității parolelor poate fi modificată sau se poate crea o rutină nouă, în schema *SYS*. Comanda *CREATE PROFILE* permite, prin intermediul parametrului *PASSWORD_VERIFY_FUNCTION*, precizarea unei funcții *PL/SQL* pentru verificarea complexității parolelor.
- Pentru fiecare utilizator, sistemul permite specificarea unei limite de folosire a resurselor disponibile, în cadrul domeniului său de securitate. Astfel, se controlează consumul resurselor importante ale sistemului. Limitarea resurselor este o metodă deosebit de utilă în cazul sistemelor *multiuser*, unde acestea sunt costisitoare. Consumul excesiv al resurselor, de către unul sau mai mulți utilizatori, poate fi în detrimentul celorlalți utilizatori ai bazei de date.
- Sistemul poate controla utilizarea resurselor la nivel de sesiune (*session level*), la nivel de apel (*call level*) sau la ambele niveluri.
- În momentul conexiunii unui utilizator la o bază de date se creează o sesiune. Fiecare sesiune consumă timp *CPU* (timp de încărcare al procesorului) și memorie. Dacă

utilizatorul depășește limita de consum a resurselor, sistemul anulează comanda curentă și returnează un mesaj prin care indică depășirea limitei de consum a resurselor. Toate comenziile anterioare tranzacției curente rămân intacte și sunt permise doar operații de tip *COMMIT*, *ROLLBACK* sau deconectare. Toate celelalte operații produc erori.

- Sistemul permite și limitarea altor resurse la nivel de sesiune: numărul de sesiuni concurente pentru fiecare utilizator, timpul în care o sesiune poate rămâne inactivă, timpul de conectare consumat pentru fiecare sesiune, spațiul *SGA* (folosit de zonele private *SQL*) al unei sesiuni etc.
- Pentru procesarea comenziilor *SQL* sau a altor tipuri de apeluri către sistem este necesar un timp *CPU*. În medie, apelurile nu sunt mari consumatoare de timp *CPU*. În schimb, o comandă *SQL* poate implica multe interogări care pot fi mari consumatoare ale acestei resurse. Pentru a preveni folosirea inadecvată a timpului *CPU*, se pot fixa limite pentru fiecare apel în parte sau pentru apelurile din cadrul unei anumite sesiuni.
- Operațiile *I/O* sunt unele dintre cele mai costisitoare operații într-un sistem de baze de date. De aceea, sistemul permite limitarea citirilor blocurilor logice de date, la nivel de apel sau de sesiune. Citirea blocurilor logice de date include citirea blocurilor de date din memorie și de pe disc. Limitările presupun numărarea blocurilor citite în timpul unui apel sau pe parcursul unei sesiuni.
- Opțiunile de limitare a resurselor care intervin în comanda *CREATE PROFILE* sunt următoarele:
 - *SESSIONS_PER_USER* (numărul maxim de sesiuni concurente);
 - *CPU_PER_SESSION* (timpul de încărcare al procesorului pentru o sesiune, exprimat în secunde);
 - *CPU_PER_CALL* (timpul de încărcare al procesorului pentru un apel, exprimat în secunde);
 - *CONNECT_TIME* (timpul total al unei sesiuni, exprimat în minute);
 - *IDLE_TIME* (timpul de inactivitate continuă pe parcursul unei sesiuni, exprimat în minute);
 - *LOGICAL_READS_PER_SESSION* (numărul de blocuri de date citite într-o sesiune, inclusiv blocurile citite din memorie sau de pe disc);
 - *LOGICAL_READS_PER_CALL* (numărul de blocuri de date citite pe parcursul unui apel pentru a procesa o comandă *SQL*);

- *PRIVATE_SGA* (dimensiunea spațiului *SGA* alocat unei sesiuni);
- *COMPOSITE_LIMIT* (costul total al resurselor pentru o sesiune).
- Înainte de crearea profilurilor și specificarea limitelor de folosire a resurselor asociate lor, trebuie să se determine valorile estimative pentru fiecare limită de resurse. Estimarea acestor limite se poate face evaluând tipul de operații pe care le execută o categorie de utilizatori. De exemplu, dacă o categorie de utilizatori, în mod normal, nu face un număr mare de citiri din blocurile logice de date, atunci valorile parametrilor referitor la acest tip de resursă pot fi mari. De obicei, cea mai bună cale de a determina valoarea aproximativă a limitei de folosire a resurselor pentru un profil al unui utilizator dat, este consultarea informațiilor istorice cu privire la ocuparea fiecărui tip de resurse.
- Pentru a obține statistici referitoare la folosirea resurselor se poate utiliza componenta de monitorizare a utilitarului *OEM* sau *SQL*Plus*. De exemplu, administratorul pentru securitate poate folosi clauza *AUDIT SESSION* pentru a obține informații despre limitele *CONNECT_TIME*, *LOGICAL_READS_PER_SESSION* și *LOGICAL_READS_PER_CALL*.
- Pentru ca profilurile să aibă efect trebuie să fie pornită opțiunea de limitare a resurselor pentru baza de date. Activarea sau dezactivarea acestei opțiuni se poate face înaintea pornirii bazei de date sau în timp ce aceasta este deschisă. În acest sens, este folosit parametrul de inițializare *RESOURCE_LIMIT* (*TRUE* pentru activare, *FALSE* pentru dezactivare).

Exemplul 13.1

Se activează opțiunea de limitare a resurselor presupunând că baza de date este deschisă.

```
ALTER SYSTEM SET RESOURCE_LIMIT = TRUE;
```

- După cum se observă, comanda *CREATE PROFILE* permite limitarea costului total al resurselor. Astfel, utilizatorul poate folosi orice resurse dorește, dar suma cantităților resurselor consumate nu poate depăși această limită. Un profil poate folosi limitări explicite de resurse, concomitent cu limitarea costului total al resurselor. Atingerea uneia dintre limite determină oprirea activității utilizatorului în sesiune. Folosirea limitărilor compuse permite mai multă flexibilitate. Valoarea corectă a unei limite compuse de resurse depinde de cantitatea totală de resurse folosită în medie de un utilizator.

- Limitările de resurse la nivel de apel (*LOGICAL_READS_PER_CALL*, *CPU_PER_CALL*) sunt impuse fiecărui apel inițiat în timpul execuției unei comenzi SQL. Depășirea uneia dintre aceste limite determină întreruperea procesării comenзii și anularea acesteia, sesiunea utilizatorului fiind menținută.
- După ce profilul a fost creat, el poate fi asociat utilizatorilor bazei de date. Nu este posibil ca un utilizator să aibă concomitent mai multe profili. Dacă un profil este atribuit unui utilizator care are deja unul, noul profil îl va înlocui pe cel vechi. Asocierea profiliurilor nu afectează sesiunea curentă. Profiliurile pot fi atribuite numai utilizatorilor, și nu unui *role* sau unui alt profil.

Exemplul 13.2

Se creează profilul *u_profil*, specificând parametrii în mod corespunzător.

```
CREATE PROFILE u_profil LIMIT
  SESSIONS_PER_USER          UNLIMITED
  CPU_PER_SESSION             UNLIMITED
  CPU_PER_CALL                1000
  CONNECT_TIME                 50
  LOGICAL_READS_PER_SESSION   DEFAULT
  LOGICAL_READS_PER_CALL      DEFAULT
  PRIVATE_SGA                  25K
  COMPOSITE_LIMIT              DEFAULT
  FAILED_LOGIN_ATTEMPTS        3
  PASSWORD_LIFE_TIME           50
  PASSWORD_REUSE_TIME          50
  PASSWORD_REUSE_MAX           DEFAULT
  PASSWORD_VERIFY_FUNCTION     DEFAULT
  PASSWORD_LOCK_TIME           1/24
  PASSWORD_GRACE_TIME          15;
```

- Folosind comanda *ALTER PROFILE* se pot modifica limitările de resurse ale unui profil. Pentru aceasta este necesar privilegiul sistem *ALTER PROFILE*. Orice utilizator care deține acest privilegiu poate modifica limitările profilului implicit. Inițial, nu există limitări de resurse în profilul implicit (sunt *UNLIMITED*). Pentru a preveni consumul nelimitat de resurse, administratorul pentru securitatea sistemului va trebui să modifice acest profil.
- Modificările făcute asupra unui profil nu vor afecta sesiunea curentă. Acestea vor deveni active pentru sesiunile ulterioare.

Exemplul 13.3

Se modifică profilul creat anterior, astfel încât după două încercări de conectare eşuate contul utilizatorului să se blocheze, parola să expire după 25 de zile și să nu se poată reutiliza aceeași parolă pentru cel puțin 60 de zile.

```
ALTER PROFILE u_profil LIMIT
  FAILED_LOGIN_ATTEMPTS      2
  PASSWORD_LIFE_TIME         30
  PASSWORD_REUSE_TIME        60
  PASSWORD_REUSE_MAX         UNLIMITED;
```

- Stergerea unui profil necesită privilegiul sistem *DROP PROFILE*. Comanda *SQL* folosită are următoarea sintaxă:

```
DROP PROFILE nume_profil [CASCADE];
```

- Opțiunea *CASCADE* se folosește dacă profilul este asociat mai multor utilizatori. Aceasta determină disocierea profilului respectiv de toți utilizatorii care îl aveau alocat. În mod automat, sistemul va atribui acestora profilul implicit.

13.1.4. Informații despre utilizatori și profili

- Sistemul menține o serie de vizualizări în dicționarului datelor care conțin informații despre utilizatorii bazei de date și despre profili:
 - *DBA_TS_QUOTAS* (descrie cotele spațiilor tabel pentru utilizatori);
 - *USER_PASSWORD_LIMITS* (descrie valorile parametrilor referitor la parole, setați prin comanda *CREATE PROFILE*);
 - *DBA_PROFILES* (listază toate profiliurile împreună cu limitele lor);
 - *USER_RESOURCE_LIMITS* (afișează limitările de resurse pentru utilizatorul curent);
 - *RESOURCE_COST* (afișează costul fiecărei resurse);
 - *V\$SESSTAT* (listază statistici despre sesiunile utilizatorilor);
 - *V\$STATNAM* (afișează numele statisticilor listate prin vizualizarea anterioară);
 - *PROXY_USERS* (descrie utilizatorii bazei de date care își pot asuma identitatea altor utilizatori) etc.

13.2. Administrarea privilegiilor și a role-urilor

13.2.1. Privilegii

- Un privilegiu este dreptul de a executa anumite comenzi *SQL*.
- Privilegiile pot include dreptul de: conectare la baza de date, creare de tabele, selectare a liniilor din tabelul altui utilizator, execuție a procedurilor stocate create de alt utilizator etc.
- Privilegiile trebuie să fie acordate utilizatorilor numai dacă sunt absolut necesare în activitatea acestora. Acordarea excesivă de privilegii poate compromite securitatea bazei de date. Privilegiile pot fi de tip sistem sau obiect.
- **Un privilegiu sistem** reprezintă dreptul de a executa anumite acțiuni asupra bazei de date sau de a grupa aceste acțiuni. De exemplu, privilegiul de a crea spații tabel sau de a șterge liniile din orice tabel al bazei sunt privilegii sistem. Există peste 100 de privilegii sistem distințe.
- Privilegiile sistem pot fi clasificate astfel:
 - privilegii specifice sistemului (de exemplu, *CREATE SESSION*, *DROP TABLESPACE*, *ALTER TABLESPACE*);
 - privilegii pentru gestiunea obiectelor create de utilizator (de exemplu, *CREATE TABLE*, *SELECT VIEW*, *EXECUTE PROCEDURE*);
 - privilegii pentru gestiunea obiectelor din orice schemă (de exemplu, *CREATE ANY TABLE*, *DROP ANY INDEX*).
- Deoarece privilegiile sistem sunt puternice, se recomandă să nu se acorde utilizatorilor obișnuiți privilegii de tip *ANY* asupra tabelelor dicționarului datelor (de exemplu, *UPDATE ANY TABLE*). Pentru a securiza accesul la dicționarul datelor, parametrul de inițializare *O7_DICTIONARY_ACCESSIBILITY* trebuie să fie *FALSE*. În acest caz, accesul la obiectele schemei *SYS* este permis doar utilizatorului care deține schema *SYS* sau care se conectează ca administrator. Privilegiile sistem acordate utilizatorilor vor permite accesul la obiectele din alte scheme, dar nu la cele din schema *SYS*.
- Pentru administratorii bazei de date există două privilegii sistem speciale, *SYSOPER* și *SYSDBA*. Aceste privilegii permit accesul la instanța bazei de date, chiar dacă baza nu este deschisă. Atunci când un utilizator având privilegiul *SYSOPER* sau *SYSDBA* se conectează la baza de date îi este asociată o schema obiectuală implicită, nu cea corespunzătoare *username*-ului său. Pentru *SYSDBA* această schema este *SYS*, iar

pentru *SYSOPER* este *PUBLIC*. Privilegiul *SYSOPER* permite folosirea comenzilor de bază, dar fără posibilitatea de a vizualiza datele utilizatorilor (*STARTUP*, *SHUTDOWN*, *ALTER DATABASE*, *CREATE SPFILE* etc.). Privilegiul *SYSDBA* permite utilizatorului să execute orice tip de operație asupra bazei de date și a obiectelor sale.

- **Privilegiile obiect** sunt drepturi de a executa anumite acțiuni asupra unui obiect specific al schemei: tabel, vizualizare, secvență, procedură, funcție, pachet. De exemplu, *ALTER TABLE*, *DELETE TABLE*, *SELECT TABLE* și *UPDATE TABLE* sunt privilegii obiect. Unele obiecte, ca de exemplu grupările, indecșii, declanșatorii, legăturile de baze de date, nu au asociate privilegii obiect. Folosirea lor este controlată de privilegiile sistem. De exemplu, pentru a modifica o grupare utilizatorul trebuie să fie proprietarul grupării sau să dețină privilegiul sistem *ALTER ANY CLUSTER*.
- În ceea ce privește respectarea privilegiilor, obiectul schemei este echivalent cu sinonimul său. Dacă se acordă un privilegiu pentru un obiect, acesta este valabil și pentru sinonimul său și reciproc. Dacă sinonimul este eliminat, toate privilegiile obiectului au în continuare efect, chiar dacă privilegiile au fost acordate doar pentru sinonimul său.

13.2.2. Role-uri

- Un *role* este un grup de privilegii înrudite care pot fi acordate sau revocate simultan utilizatorilor sau altor *role*-uri;
- Folosirea *role*-urilor permite:
 - simplificarea administrării privilegiilor (în loc să se acorde mai multe privilegii în mod explicit unui grup de utilizatori înrudiți, se poate crea un *role* care să conțină toate privilegiile necesare și apoi să se acorde acest *role* fiecărui membru al grupului);
 - administrarea dinamică a privilegiilor (dacă privilegiile unui grup de utilizatori trebuie schimbate, modificarea lor se va face la nivelul *role*-ului care le conține și aceasta se propagă automat pentru fiecare utilizator care are asociat *role*-ul respectiv);
 - activarea selectivă a privilegiilor (*role*-urile pot fi activate sau dezactivate selectiv, astfel încât se permite un control ridicat al privilegiilor acordate utilizatorilor).

- Se pot crea aplicații care să interogheze dicționarul datelor și să activeze/dezactiveze automat unele *role-uri*, în funcție de utilizatorul care încearcă să execute aplicația respectivă.
- *Role-urile* au următoarele caracteristici:
 - pot fi acordate sau revocate utilizatorilor folosind aceleași comenzi ca și în cazul privilegiilor sistem;
 - pot cuprinde atât privilegii sistem, cât și privilegii obiect;
 - pot fi protejate prin folosirea parolelor;
 - trebuie să aibă un nume unic, diferit de conturile utilizatorilor și de celealte *role-uri* din baza de date;
 - nu sunt conținute în schema nici unui utilizator;
 - caracteristicile lor pot fi regăsite în dicționarul datelor.
- La crearea bazei de date *Oracle* se definesc automat *role-uri* predefinite. Acestea li se pot acorda sau revoca alte privilegii sau *role-uri*. Un exemplu de *role* predefinit este *CONNECT* care include privilegiile sistem *CREATE SESSION*, *ALTER SESSION*, *CREATE TABLE* etc.
- Pentru a permite utilizatorilor care nu dețin un *role DBA* accesul la vizualizările dicționarului datelor, sistemul oferă *role-urile* *DELETE_CATALOG_ROLE*, *EXECUTE_CATALOG_ROLE* și *SELECT_CATALOG_ROLE*.
- Crearea unui *role* necesită privilegiul sistem *CREATE ROLE* și se realizează prin comanda cu același nume. După creare, *role-ul* nu are privilegii sau *role-uri* asociate, acestea putând fi acordate ulterior.
- Comanda prin care se creează un *role* are următoarea sintaxă:

```
CREATE ROLE nume_role
  { NOT IDENTIFIED | IDENTIFIED tip_autorizare };
```

- Dacă este folosită clauza *NOT IDENTIFIED* atunci nu se cere nici o autorizare pentru *role-ul* respectiv. Această opțiune este implicită.
- Clauza *tip_autorizare* specifică următoarelor categorii de autorizări:
 - prin baza de date (*IDENTIFIED BY parolă*);
 - prin intermediul unei aplicații care utilizează pachete specifice (*IDENTIFIED USING [schema.]nume_pachet*);
 - externă, realizându-se prin sistemul de operare, rețea sau alte surse externe (*IDENTIFIED EXTERNALLY*);

- globală (*IDENTIFIED GLOBALLY*).
- Dacă autorizarea *role*-ului se face prin sistemul de operare, atunci informațiile pentru fiecare utilizator trebuie configurate la acest nivel. În cazul autorizării prin rețea, dacă utilizatorii se conectează la baza de date prin *Oracle Net*, atunci *role*-urile nu pot fi autorizate de sistemul de operare.
- Autorizarea globală presupune existența *role*-urilor globale. Un *role* global se definește în baza de date, dar nu poate fi acordat de la acest nivel altui utilizator sau *role*. Atunci când un utilizator global încearcă să se conecteze la bază este interrogat directorul de servicii pentru a se obține *role*-urile globale asociate.
- Schimbarea modului de autorizare a unui *role* se face folosind comanda *ALTER ROLE*. Pentru aceasta este necesar privilegiul sistem *ALTER ANY ROLE* sau un *role* cu opțiuni de administrare.
- Unui utilizator îi pot fi acordate mai multe *role*-uri. La fiecare conectare a utilizatorului este activat în mod automat un *role* implicit, chiar dacă acesta nu are asociat explicit nici un *role*. Un *role* implicit este specificat folosind clauza *DEFAULT ROLE* din comanda *ALTER USER*:

```
ALTER USER nume_utilizator DEFAULT ROLE
    {role [, role ...] |
     ALL [EXCEPT role [, role ...] ] | NONE};
```

- Folosind opțiunea *ALL*, toate *role*-urile acordate utilizatorului devin implicite, cu excepția celor care apar în clauza *EXCEPT*.
- Opțiunea *NONE* este utilizată dacă utilizatorul nu trebuie să dețină *role*-uri implicite. În acest caz, la conectare utilizatorul va beneficia doar de privilegiile care i-au fost acordate în mod explicit.
- Clauza *DEFAULT ROLE* nu poate fi folosită pentru *role*-luri:
 - care nu au fost acordate utilizatorului;
 - acordate prin intermediul altor *role*-uri;
 - administrate prin servicii externe.
- Deoarece *role*-urile trebuie inițial acordate unui utilizator și apoi declarate implicite, nu se pot seta *role*-uri implicite folosind comanda *CREATE USER*.
- Atunci când un *role* este activ, utilizatorul poate folosi privilegiile acordate prin intermediul acestuia. Dacă *role*-ul este inactiv, atunci utilizatorul poate folosi doar privilegiile care i-au fost acordate în mod explicit sau care fac parte din alte *role*-uri deținute de acesta. Activarea sau dezactivarea *role*-urilor persistă doar pentru sesiunea

currentă. În sesiunile următoare, utilizatorul va avea activate din nou *role*-urile implicate.

- Pentru activarea sau dezactivarea *role*-urilor este utilizată comanda *SET ROLE*. Aceasta are următoarea sintaxă:

```
SET ROLE {role [IDENTIFIED BY parolă]
           [, role [IDENTIFIED BY parolă]...]
           | ALL [EXCEPT role [, role ...] ] | NONE};
```

- Clauza *IDENTIFIED BY* precizează parola necesară pentru autorizarea *role*-ului.
- Folosind opțiunea *ALL* sunt activate toate *role*-urile acordate utilizatorului curent, cu excepția celor care apar în clauza *EXCEPT*.
- Opțiunea *NONE* asigură dezactivarea tuturor *role*-urilor care erau acordate utilizatorului respectiv.
- În unele cazuri este necesară suprimarea unui *role* din baza de date. Ștergerea acestuia implică suprimarea sa din toate *role*-urile cărora le-a fost acordat. Deoarece crearea obiectelor nu este dependentă de privilegiile primite prin *role*-uri, tabelele și celealte obiecte nu vor fi șterse în momentul suprimării *role*-ului. Comanda *SQL* de ștergere a unui *role* este *DROP ROLE*. Pentru folosirea acestei comenzi este necesar privilegiul sistem *DROP ANY ROLE* sau un *role* cu opțiuni de administrare.

13.2.3. Acordarea privilegiilor și a *role*-urilor

- Privilegiile și *role*-urile pot fi acordate utilizatorilor sau altor *role*-uri folosind comanda *GRANT* sau utilitarul *Oracle Enterprise Manager*.
- Nu pot acorda privilegii sistem sau *role*-uri decât utilizatorii cărora le-au fost acordate acestea cu opțiunea *WITH ADMIN OPTION* a comenzi *GRANT* sau cei care dețin privilegiul sistem *GRANT ANY ROLE*.
- Utilizatorii nu pot acorda *role*-uri autorizate global. Acordarea, respectiv revocarea unui astfel de *role* este controlată în întregime prin directorul de servicii.
- Comanda *SQL* prin care se acordă privilegii sistem sau *role*-uri unui utilizator are următoarea sintaxă:

```
GRANT {privilegiu_sistem | role} [, {privilegiu_sistem | role}...]
       TO {nume_utilizator | role | PUBLIC}
            [, {nume_utilizator | role | PUBLIC}...]
            [WITH ADMIN OPTION];
```

- Opțiunea *PUBLIC* permite acordarea privilegiilor sistem tuturor utilizatorilor bazei de date.
- Clauza *WITH ADMIN OPTION* permite utilizatorilor să acorde mai departe privilegiile sistem și *role*-urile respective altor utilizatori sau *role*-uri.
- Privilegiile obiect nu pot fi acordate împreună cu privilegii sistem sau cu *role*-uri, în cadrul aceleiași comenzi *GRANT*. Acordarea de privilegii obiect utilizatorilor sau *role*-urilor poate fi făcută de proprietarul obiectului sau de un utilizator căruia i s-a acordat privilegiul obiect respectiv, cu opțiunea *WITH GRANT OPTION*.
- Comanda folosită pentru acordarea unui privilegiu obiect este:

```
GRANT {privilegiu_object [ (listă_coloane) ]
        [, privilegiu_object [ (listă_coloane)...]
         | ALL [PRIVILEGES] ]
ON [nume_schemă.]obiect
TO {nume_utilizator | role | PUBLIC}
     [, {nume_utilizator | role | PUBLIC}...]
[WITH GRANT OPTION];
```

- Prin opțiunea *listă_coloane* se precizează coloanele unui tabel sau ale unei vizualizări pentru care se acordă privilegiul.
- Această opțiune poate fi folosită atunci când sunt acordate privilegii obiect de tip *INSERT*, *REFERENCES* sau *UPDATE*.
- Opțiunea *ALL* permite acordarea tuturor privilegiilor obiect pentru care utilizatorul ce inițiază comanda *GRANT* are opțiunea *WITH GRANT OPTION*. Clauza *ON [nume_schemă.]obiect* precizează obiectul relativ la care este acordat privilegiul. Opțiunea *PUBLIC* permite acordarea privilegiilor obiect tuturor utilizatorilor bazei de date.
- Clauza *WITH GRANT OPTION* permite utilizatorilor să acorde privilegii obiect altor utilizatori sau *role*-uri. Această clauză nu poate fi utilizată atunci când privilegiul obiect este acordat unui *role*.
- În ceea ce privește privilegiile referitoare la comenzi *LMD*, acestea sunt acordate pentru operațiile *DELETE*, *INSERT*, *SELECT* și *UPDATE* asupra unui tabel sau unei vizualizări, doar utilizatorilor sau *role*-urilor care trebuie să interogheze sau să prelucreze datele respective.
- Privilegiile *INSERT* și *UPDATE* se pot restricționa pentru anumite coloane ale tabelului. În cazul unui privilegiu *INSERT* restricționat pentru anumite coloane, inserarea unei linii permite inserarea de valori doar pentru coloanele accesibile.

- Coloanele restricționate primesc valori implicate sau *null*. În cazul unui privilegiu *UPDATE* restricționat, vor putea fi modificate doar coloanele pentru care utilizatorul are acest drept.
- Privilegiile *ALTER*, *INDEX* și *REFERENCES* permit executarea de operații *LDD* asupra unui tabel. Un utilizator care încearcă să execute o comandă *LDD* trebuie să aibă anumite privilegii sistem sau obiect. De exemplu, pentru a crea un declanșator asupra unui tabel, utilizatorul trebuie să dețină privilegiul obiect *ALTER TABLE* și privilegiul sistem *CREATE TRIGGER*. Privilegiul *REFERENCES* poate fi acordat unei anumite coloane a unui tabel. Astfel, tabelul respectiv este folosit ca tabel „părinte“ pentru orice cheie externă care trebuie creată.

Exemplul 13.4

- a) Se acordă utilizatorului *u1* dreptul de a porni și opri baza de date, fără a i se permite crearea unei baze de date.

```
GRANT SYSOPER TO u1;
```

- b) Se acordă utilizatorilor *u2* și *u3* privilegiile obiect *SELECT* și *INSERT* asupra coloanelor tabelului *produse*, cu posibilitatea ca aceștia să acorde privilegiile și altor utilizatori sau *role-uri*.

```
GRANT SELECT, INSERT ON produse TO u2, u3
WITH GRANT OPTION;
```

- c) Se acordă utilizatorului *u4* privilegiul obiect *INSERT* doar pentru coloanele *id_produs* și *denumire*.

```
GRANT INSERT (id_produs, denumire) ON produse TO u4;
```

Exemplul 13.5

- a) Se creează un *role* numit *u_role*, care să permită utilizatorilor să creeze tabele și vizualizări.

```
CREATE ROLE u_role;
GRANT CREATE TABLE, CREATE VIEW TO u_role;
```

- b) Se atribuie *role-ul* creat anterior și *role-ul* predefinit *RESOURCE*, utilizatorului *u5*. *Role-ul RESOURCE* trebuie să fie activat în mod automat atunci când utilizatorul se conectează.

```
GRANT user_role, RESOURCE TO u5;
ALTER USER u1
DEFAULT ROLE RESOURCE;
```

- c) Se permite utilizatorului *u5* să consulte vizualizările dicționarului.

```
GRANT SELECT_CATALOG_ROLE TO u5;
```

- d) Se afișează segmentele *undo* care sunt utilizate de instanța curentă (se presupune că utilizatorul curent este *u5*).

```
SET ROLE SELECT_CATALOG_ROLE;
SELECT SEGMENT_NAME
FROM   DBA_ROLLBACK_SEGS
WHERE  STATUS = 'ONLINE';
```

13.2.4. Revocarea privilegiilor și a *role*-urilor

- Pentru a revoca privilegii sau *role*-uri se poate folosi comanda *REVOKE* sau utilitarul *Oracle Enterprise Manager*.
- Un utilizator care are opțiunea de administrare, de acordare a unui privilegiu sau *role*, le poate revoca pe acestea oricărui *role* sau utilizator al bazei de date. Un utilizator care deține privilegiul *GRANT ANY ROLE* poate revoca orice *role*.
- Sintaxa generală a comenzi de revocare a unui privilegiu sistem sau *role* este următoarea:

```
REVOKE {privilegiu_sistem | role}
      [, {privilegiu_sistem | role}...]
  FROM    {nume_utilizator | role | PUBLIC};
```

- Opțiunea *PUBLIC* permite revocarea privilegiilor sistem sau a *role*-urilor tuturor utilizatorilor bazei de date.
- Sintaxa generală a comenzi de revocare a unui privilegiu obiect este:

```
REVOKE {privilegiu_object [, privilegiu_object ...]
          | ALL [PRIVILEGES] }
  ON [nume_schemă.]objec
  FROM {nume_utilizator | role | PUBLIC}
        [, {nume_utilizator | role | PUBLIC}...]
  [CASCADE CONSTRAINTS];
```

- Opțiunea *ALL* permite revocarea tuturor privilegiilor obiect acordate utilizatorului.
- Prin clauza *ON* se identifică obiectul la care se referă privilegiul ce trebuie revocat.
- Clauza *FROM* identifică utilizatorii sau *role*-urile pentru care este revocat privilegiul obiect.
- Opțiunea *PUBLIC* determină revocarea unor privilegii obiect tuturor utilizatorilor bazei de date.
- Clauza *CASCADE CONSTRAINTS* permite suprimarea constrângerilor de integritate referențială definite folosindu-se privilegiile *REFERENCES* sau *ALL*.

- Definițiile obiectelor care depind de privilegii *LMD* sistem sau obiect pot fi afectate dacă privilegiile respective sunt revocate.
 - De exemplu, dacă privilegiul sistem *SELECT ANY TABLE* a fost acordat unui utilizator care apoi a creat proceduri sau vizualizări ce folosesc un tabel din altă schemă, atunci revocarea acestui privilegiu determină invalidarea procedurilor sau vizualizărilor respective.
 - De asemenea, dacă o procedură include comenzi *SQL* prin care sunt selectate datele unui tabel și este revocat privilegiul obiect *SELECT* asupra tabelului respectiv, atunci procedura nu se mai execută cu succes.
- Definițiile obiectelor pentru care sunt necesare privilegiile obiect *ALTER* și *INDEX* nu sunt afectate dacă aceste privilegii sunt revocate.
 - De exemplu, dacă privilegiul *INDEX* este revocat unui utilizator care a creat un index asupra unui tabel al altui utilizator, indexul respectiv va continua să existe și după revocare.
- Revocarea unui privilegiu obiect poate determina efectul de revocare în cascadă a acestuia.
 - De exemplu, dacă utilizatorului *u1* i se acordă privilegiul obiect *SELECT* asupra unui tabel, cu opțiunea *WITH GRANT OPTION*, iar acesta îl acordă utilizatorului *u2*, atunci revocarea privilegiului utilizatorului *u1* va determina automat revocarea acestui privilegiu și pentru utilizatorul *u2*.

13.2.5. Informații despre privilegii și role-uri

- Sistemul menține o serie de vizualizări în dicționarului datelor, care conțin informații despre privilegii și *role-uri*:
 - *DBA_SYS_PRIVS* (privilegiile sistem acordate utilizatorilor sau altor *role-uri*);
 - *SESSION_PRIVS* (privilegiile sistem și obiect disponibile utilizatorului în sesiunea curentă);
 - *DBA_TAB_PRIVS* (privilegiile acordate asupra obiectelor bazei);
 - *DBA_COL_PRIVS* (coloanele obiectelor care sunt referite în privilegii);
 - *DBA_ROLES* (*role-urile definite* în baza de date);
 - *DBA_ROLES_PRIVS* (*role-urile acordate* utilizatorilor sau altor *role-uri*);
 - *ROLE_ROL_PRIVS* (*role-urile acordate* altor *role-uri*);
 - *SESSION_ROLES* (*role-urile active* pentru utilizator în sesiunea curentă);

- *ROLE_SYS_PRIVS* (privilegiile sistem acordate *role*-urilor);
- *ROLE_TAB_PRIVS* (privilegiile obiect acordate *role*-urilor) etc.

13.4. Auditarea

- Auditarea presupune monitorizarea și înregistrarea selectivă a acțiunilor utilizatorilor unei baze de date. Auditarea este un proces folosit pentru:
 - investigarea activităților suspecte (de exemplu, dacă un utilizator neautorizat șterge date din tabele, administratorul pentru securitate trebuie să verifice toate conexiunile la bază și operațiile de ștergere a liniilor din tabelele bazei de date pentru a-l identifica);
 - monitorizarea și gruparea datelor pe categorii de activități specifice în cadrul bazei de date (de exemplu, administratorul bazei trebuie să colecteze statistici despre tabelele care au fost modificate, numărul de operații *I/O* executate, durata medie a unei sesiuni, privilegiile sistem folosite, numărul de utilizatori care s-au conectat simultan la diferite intervale de timp etc.).

13.4.1. Opțiuni de audit

- Controlul acțiunilor întreprinse asupra elementelor unei baze de date este realizat prin comanda *AUDIT*, iar rezultatele verificărilor sunt înregistrate într-un tabel (*AUD\$*) al dicționarului datelor, cunoscut sub denumirea de *audit trail*. Pentru a preveni completarea totală a spațiului tabel asociat, periodic se șterg înregistrări din tabelul *AUD\$*. Se recomandă ca tabelul *AUD\$* să nu aparțină spațiului tabel *SYSTEM*. De asemenea, acesta trebuie protejat împotriva accesului neautorizat.
- Sintaxa simplificată a comenzii *AUDIT* este următoarea:

```
AUDIT { comandă_SQL [,comandă_SQL ...]
        | privilegiu_sistem, [, privilegiu_sistem ...]
          [BY utilizator]
          | ON [nume_schemă.]obiect}
[BY {SESSION | ACCESS} ]
[WHENEVER [NOT] SUCCESSFUL];
```

- Sistemul *Oracle* suportă trei tipuri generale de audit, la nivel de comandă, privilegiu sau obiect ale unei scheme. Operațiile de audit pot fi definite la nivel de sesiune sau acces. Pentru a dezactiva opțiunile de audit inițiate printr-o comandă *AUDIT* se utilizează comanda *NOAUDIT* asupra celor opțiuni.

- Auditul la nivel de comandă se referă la verificări selective asupra comenziilor *SQL*, care se găsesc în una dintre următoarele două categorii:
 - comenzi *LDD* relativ la un anumit obiect al schemei (de exemplu, *AUDIT TABLE* verifică toate comenziile *CREATE* și *DROP TABLE*);
 - comenzi *LMD* referitoare la anumite obiecte ale schemei, dar fără să se specifiche numele acestora (de exemplu, *AUDIT SELECT TABLE* controlează toate operațiile *SELECT* asupra tabelelor și vizualizărilor).
- Sistemul *Oracle* permite verificarea selectivă a execuțiilor comenziilor *SQL*. Execuțiile eşuate se pot verifica doar dacă structura *SQL* este validă, iar eşuarea s-a produs din cauza referirii unui obiect inexistent sau a lipsei unei autorizații adecvate. Comenziile care eşuează pentru că nu au fost valide, nu pot fi verificate.
- Pentru verificarea exclusivă a execuțiilor cu succes se utilizează clauza *WHenever SUCCESSFUL* a comenzi *AUDIT*, iar pentru verificarea doar a execuțiilor eşuate clauza *WHenever NOT SUCCESSFUL* a aceleiași comenzi. Dacă nu este folosită nici una dintre clauzele menționate mai sus, verificarea se realizează în ambele cazuri.
- Auditul la nivel de privilegii asigură verificarea selectivă a comenziilor care necesită privilegii sistem.
 - De exemplu, verificarea privilegiului sistem *SELECT ANY TABLE* permite monitorizarea comenziilor care se execută folosind acest privilegiu.
- Privilegiile obiect sunt verificate înaintea privilegiilor sistem. Dacă sunt setate opțiuni de audit pentru comenzi și privilegii similare, atunci este generată o singură înregistrare pentru audit.
 - De exemplu, dacă comanda *CREATE TABLE* și privilegiul sistem *CREATE TABLE* sunt ambele verificate, atunci se generează o singură înregistrare de audit, de fiecare dată când este creat un tabel.
- Auditul la nivel de obiect al schemei constă în verificarea comenziilor *LMD* specifice și a comenziilor *GRANT* sau *REVOKE* pentru obiectele schemei.
- Se pot verifica acele comenzi care referă tabele, vizualizări, secvențe, proceduri stocate, funcții, pachete. P
 - Procedurile din pachete nu pot fi verificate individual.
 - De asemenea, comenziile care referă grupări, indecsi sau sinonime, nu pot fi verificate direct.

- Totuși, se poate verifica accesul la aceste obiecte în mod indirect, prin verificarea operațiilor care afectează tabelul de bază.

Exemplul 13.6

Se consideră următoarea secvență de comenzi *SQL*:

```
AUDIT SELECT ON produse;

CREATE VIEW produse_categorii AS
    SELECT p.id_produs, p.denumire, c.denumire, c.nivel
    FROM produse p, categorii c
    WHERE p.id_produs = c.id_produs;

AUDIT SELECT ON produse_categorii;

SELECT * FROM produse_categorii;
```

Ca urmare a interogării vizualizării *produse_categorii* sunt generate două înregistrări de auditare, corespunzătoare interogării vizualizării *produse_categorii*, respectiv tabelului de bază *produse*. Interrogarea tabelului de bază *categorii* nu generează înregistrări de verificare, deoarece opțiunea de *AUDIT SELECT* pentru acest tabel nu este activată.

- Opțiunile de audit pot fi specificate la nivel de sesiune sau acces, folosind clauzele *BY SESSION*, respectiv *BY ACCESS* ale comenzi *AUDIT*.
- Auditarea la nivel de sesiune are ca rezultat inserarea unei singure înregistrări în tabelul *AUD\$* pentru fiecare utilizator și obiect al schemei, în timpul sesiunii care include o acțiune auditată.
- Acțiunea de audit la nivel de acces presupune inserarea unei înregistrări în tabelul *AUD\$* pentru fiecare execuție a unei operații care este monitorizată.
 - Operațiile de audit la nivel de comenzi sau de privilegii, aplicate pentru comenzi *LDD* nu pot fi precizate decât la nivel de acces.
- Opțiunile de audit la nivel de comenzi sau privilegii permit ca monitorizarea să se realizeze pentru un anumit utilizator sau pentru un grup de utilizatori.
 - Prin utilizarea operației de audit asupra unu grup de utilizatori se poate minimiza numărul de înregistrări din tabelul *AUD\$*.

Exemplul 13.7

- a) Se dispune auditarea comenzii *SELECT TABLE* folosite de utilizatorii *u1* și *u2*.

```
AUDIT SELECT TABLE
BY U1, U2;
```

- b) Se presupune că:

- auditarea se realizează la nivel de sesiune pentru toate comenzi *SELECT TABLE*;
- utilizatorul *u1* se conectează la baza de date, execută trei comenzi *SELECT* asupra tabelului *produse* și apoi se deconectează;
- utilizatorul *u2* se conectează la baza de date, execută două comenzi *SELECT* asupra tabelului *categorii* și apoi se deconectează.

În tabelul *AUD\$* se vor insera două înregistrări, câte una pentru fiecare sesiune ce conține comenzi auditate.

Dacă același utilizator execută interogări asupra unor tabele distințe, atunci în tabelul *AUD\$* se vor insera mai multe înregistrări, câte una pentru fiecare tabel.

- c) Se presupune că:

- auditarea se realizează la nivel de acces pentru toate comenzi *SELECT TABLE*;
- utilizatorul *u1* se conectează la baza de date, execută trei comenzi *SELECT* asupra tabelului *produse* și apoi se deconectează;
- utilizatorul *u2* se conectează la baza de date, execută patru comenzi *SELECT* asupra tabelului *produse* și apoi se deconectează.

În tabelul *AUD\$* se vor insera șapte înregistrări, câte una pentru fiecare comandă *SELECT*.

13.4.2. Mecanisme pentru audit

- Înregistrarea informațiilor pentru audit poate fi activată sau dezactivată. Dacă opțiunea de auditare este activă, atunci se permite oricărui utilizator autorizat să specifiche opțiuni pentru audit, rezervând administratorului pentru securitate dreptul de control asupra înregistrării informațiilor de audit.
- Informațiile de audit conțin numele contului, identificatorul sesiunii, identificatorul mașinii *client*, numele schemei de obiecte care a fost accesată, operațiile executate sau inițiate, codul rezultat în urma compilării operației, privilegiile sistem folosite etc.
- Activarea sau dezactivarea operațiilor de audit pentru o instanță se realizează prin parametrul de initializare *AUDIT_TRAIL*. Acest parametru poate avea una din următoarele valori:

- *TRUE* sau *DB* (se activează operațiile de audit și informațiile despre acestea sunt înregistrate în tabelul *AUD\$* al bazei de date);
 - *OS* (se activează operațiile de audit și informațiile despre acestea sunt înregistrate în tabelul *audit trail* al sistemului de operare);
 - *FALSE* sau *NONE* (se dezactivează operațiile de audit).
- Atunci când opțiunea de audit devine activă se generează verificări asupra înregistrărilor pe durata derulării fazelor de execuție a comenzi. Comenzile *SQL* din interiorul unităților de program *PL/SQL* sunt verificate individual.
 - Chiar dacă auditul bazei de date nu este activat, sistemul *Oracle* înregistrează întotdeauna în tabelul *audit trail* al sistemului de operare, informații despre următoarele acțiuni care au loc asupra bazei:
 - pornirea instanței (utilizatorul care a pornit instanța, identificatorul mașinii *client*, data și momentul de timp etc.);
 - oprirea instanței (utilizatorul care a închis instanța, identificatorul mașinii *client*, data și momentul de timp etc.);
 - sesiunile utilizatorilor cu privilegii de administrare.
 - Opțiunile de audit la nivel de comenzi *SQL* sau privilegii au efect pe timpul conexiunii unui utilizator la baza de date (pe toată durata sesiunii). Schimbările opțiunilor de audit la nivel de comenzi și privilegii nu afectează sesiunea curentă. Modificările au efect numai asupra noilor sesiuni inițiate. În schimb, modificările opțiunilor de audit la nivel de obiecte ale schemei au efect imediat, inclusiv pentru sesiunea curentă.
 - Operația de audit a bazei de date nu permite monitorizări la nivel de coloană. Dacă acest lucru este necesar, atunci se pot folosi rutine speciale pentru auditare (de exemplu, proceduri stocate, declanșatori). Baza de date permite mecanisme de audit integrate, astfel încât monitorizările pot fi realizate automat, fără intervenția utilizatorilor.
 - Sistemul *Oracle* oferă procedee de auditare granulară (*fine-grained auditing*) prin care se permite monitorizarea accesului la date. Administratorul pentru securitate poate utiliza pachetul *DBMS_FGA* cu scopul de a crea politici de audit pentru un anumit tabel.
 - Dacă fiecare linie returnată de o interogare îndeplinește condițiile de audit, atunci se înregistrează în tabelul *AUD\$* un eveniment de audit care include *username*-ul, codul *SQL*, variabilele de legătură, identificatorul sesiunii, momentul de timp etc.

- Opțional, administratorii pot defini modalități de prelucrare și procesare a evenimentelor de audit. De exemplu, la prelucrarea unui eveniment de audit se pot trimite mesaje de avertizare administratorului. Toate informațiile relevante sunt livrate printr-un mecanism asemănător declanșatorilor.

13.4.3. Informații despre audit

- Informații despre opțiunile de audit și evenimentele de audit înregistrate se pot obține consultând dicționarul datelor. Dintre cele mai importante vizualizări referitoare la auditul bazei de date se remarcă:
 - *ALL_DEF_AUDIT_OPTS* (opțiunile implicite pentru audit);
 - *DBA_STMT_AUDIT_OPTS* (opțiunile de audit la nivel de comandă);
 - *DBA_PRIV_AUDIT_OPTS* (opțiunile de audit la nivel de privilegiu);
 - *DBA_OBJ_AUDIT_OPTS* (opțiunile de audit la nivel de obiect);
 - *DBA_AUDIT_TRAIL* (toate înregistrările din *audit trail*);
 - *DBA_AUDIT_OBJECT* (înregistrările din *audit trail* referitoare la obiectele schemei);
 - *DBA_AUDIT_SESSION* (înregistrările din *audit trail* pentru operațiile de audit la nivel de sesiune);
 - *DBA_AUDIT_STATEMENT* (înregistrările din *audit trail* referitoare la comenzi etc.

Bibliografie

1. Connolly T.M., Begg C.E., Database Systems: *A Practical Approach to Design, Implementation and Management*, 5th edition, Pearson Education, 2005
2. Dollinger R., Andron L., *Baze de date și gestiunea tranzacțiilor*, Editura Albastră, Cluj-Napoca, 2004
3. Oracle and/or its affiliates, *Oracle Database Concepts*, 1993, 2025
4. Oracle and/or its affiliates, *Oracle Database Performance Tuning Guide*, 2013, 2025
5. Oracle and/or its affiliates, *Oracle Database SQL Language Reference*, 1996, 2025
6. Oracle and/or its affiliates, *Oracle Database PL/SQL Language Reference*, 1996, 2025
7. Oracle and/or its affiliates, *Oracle Database Administrator's Guide*, 2001, 2023
8. Oracle University, *Oracle Database: PL/SQL Fundamentals, Student Guide*, 2009, 2025
9. Popescu I., Alecu A., Velcescu L., Florea (Mihai) G., *Programare avansată în Oracle9i*, Ed. Tehnică, 2004